Self-Embeddedness

**Theorem**

Let $G$ be a CFG in Chomsky Normal Form. Let us call the production of the form:

\[
\text{Nonterminal} \rightarrow \text{Nonterminal Nonterminal}
\]

**live** and the productions of the form

\[
\text{Nonterminal} \rightarrow \text{terminal}
\]

**dead**. If we restrict to using live productions at most once each, we can generate only finitely many words.
Self-Embeddedness

• Every time we apply a **live** production, we increase the number of nonterminals by one
• Every time we apply a **dead** production, we decrease the number of nonterminals by one
• We will always apply one more **dead** production than **live** productions.
• Show the *self-embeddedness* of any word generated by $S$

**Example**

\[
\begin{align*}
S & \rightarrow AZ \\
Z & \rightarrow BB \\
B & \rightarrow ZA \\
A & \rightarrow a \\
B & \rightarrow b
\end{align*}
\]
Self-Embeddedness

Note

When we expand the productions of a grammar in CNF, we will always produce a **binary tree** as our *derivation tree*.

Because of this property, we can theoretically construct a *complete* binary tree.
Self-Embeddedness

Note

When we expand the productions of a grammar in CNF, we will always produce a **binary tree** as our **derivation tree**.

Because of this property, we can theoretically construct a **complete** binary tree

Theorem

*If $G$ is a CFG in CNF that has $p$ live productions and $q$ dead productions, and if $w$ is a word generated by $G$ that has more than $2^p$ letters in it, then somewhere in every derivation tree for $w$ there is an example of some nonterminal (call it $Z$) being used twice where the second $Z$ is descended from the first $Z$.***
Self-Embeddedness

Note

When we expand the productions of a grammar in CNF, we will always produce a **binary tree** as our **derivation tree**.

Because of this property, we can theoretically construct a **complete** binary tree.

Theorem

If $G$ is a CFG in CNF that has $p$ live productions and $q$ dead productions, and if $w$ is a word generated by $G$ that has more than $2^p$ letters in it, then somewhere in every derivation tree for $w$ there is an example of some nonterminal (call it $Z$) being used twice where the second $Z$ is descended from the first $Z$.

The **live** productions indicate the maximum **depth** of the tree.
Self-Embeddedness

Definition

In a given derivation of a word in a given CFG, a nonterminal is said to be self-embedded if it ever occurs as a tree descendent of itself.

Example

CFG for NONNULLPALINDROME — derivation for aabaa

\[
\begin{align*}
S & \rightarrow AX \\
X & \rightarrow SA \\
S & \rightarrow BY \\
Y & \rightarrow SB \\
S & \rightarrow a \\
S & \rightarrow b \\
S & \rightarrow AA \\
S & \rightarrow BB \\
A & \rightarrow a \\
B & \rightarrow b
\end{align*}
\]
Self Embeddedness

**Definition**

Let us introduce the notation $\ast \Rightarrow$ to stand for the phrase “can eventually produce”. It is used in the following context:

Suppose in a certain CFG the working string $S_1$ can produce the working string $S_2$, which in turn can produce $S_3 \ldots S_n$

We can then write:

$$S_1 \Rightarrow S_n$$
Self Embeddedness

Definition

Let us introduce the notation \( \Rightarrow^* \) to stand for the phrase “can eventually produce”. It is used in the following context:

Suppose in a certain CFG the working string \( S_1 \) can produce the working string \( S_2 \), which in turn can produce \( S_3 \ldots S_n \).

We can then write:

\[
S_1 \Rightarrow^* S_n
\]

For NONNULLPALINDROME, we can state the following:

\[
X \Rightarrow^* a^n X a^n
\]
Non-Context-Free Languages

- It turns out that not all languages are context-free.
- The simplest example of a non-context-free language is

  $$\{a^n b^n c^n \mid n \geq 0\}.$$

- To process this would require two stacks.
The Pumping Lemma

Theorem (The Pumping Lemma for Context-Free Grammars)

If \( L \) is a context-free language then there exists an integer \( p \) such that if any string \( s \in L \) has length at least \( p \), then \( s \) may be divided into five substrings \( s = uvxyz \) such that

- \(|vy| > 0\),
- \(|vxy| \leq p\),
- \(uv^i xy^i z \in L\) for all \( i \geq 0\).
The Pumping Lemma Parts

- $u$ — the substring of all the letters of $w$ generated to the “left” of the derivation we care about
- $v$ — the substring of all the letters of $w$ descended from the root of the derivation we care about but to the left of the **self-embedded** state
- $x$ — the substring of all the letters of $w$ descended from the **self-embedded** state
- $y$ — the substring of all the letters of $w$ descended from the right of the **self-embedded** state to the end of the derivation we care about
- $z$ — the substring of all the letters of $w$ generated to the “right” of the derivation we care about
The proof is somewhat similar to the proof of the Pumping Lemma for Regular Languages except that it is based on a grammar rather than a machine.

But first, an example...
An Example

Example

• Let $L = \{a^n b^n c^n \mid n \geq 0\}$.
• We will use the Pumping Lemma to show that $L$ is not context-free.
An Example

Proof.

- Suppose $L$ is context-free.
- Then let $p$ be the “pumping length” of $L$ (for CFLs).
- Let $s = a^p b^p c^p$.
- Then $s = uvxyz$ such that $|vy| > 0$, $|vxy| \leq p$, and $uv^i xy^i z \in L$.
- We will show that this is not possible.
An Example

Proof.

• $vxy$ is the “middle part” of $uvxyz$ and it has length at most $p$.
• Therefore, it consists of
  Case 1: All $a$’s,
  Case 2: Some $a$’s followed by some $b$’s,
  Case 3: All $b$’s,
  Case 4: Some $b$’s followed by some $c$’s, or
  Case 5: All $c$’s.

□
An Example

Proof.

• It is enough to consider the first two cases.
• The other three cases are similar.
• Case 1: Suppose \(vxy\) consists of all \(a\)'s.
  • Then \(v = a^k\) and \(y = a^m\) for some \(k, m\), not both 0.
  • So \(uv^2xy^2z = a^{p+k+m}b^{p}c^{p}\), which is not in \(L\).
  • This is a contradiction.
An Example

Proof.

- Case 2: \( vxy \) consists of some \( a \)'s followed by some \( b \)'s.
  - There are three possibilities:
    - \( v \) is all \( a \)'s and \( y \) is all \( b \)'s,
    - \( v \) is all \( a \)'s and \( y \) is some \( a \)'s followed by some \( b \)'s,
    - \( v \) is some \( a \)'s followed by some \( b \)'s and \( y \) is all \( b \)'s.

\( \square \)
An Example

Proof.

• Case 2, continued…
  • It doesn’t really matter which is the case because both $v$ and $y$ get pumped up.
  • Let $k$ be the number of $a$’s and $m$ be the number of $b$’s altogether in $vy$, $m$ and $k$ are not both 0 (but possibly $m = k$).
  • So $uv^2xy^2z$ will contain $p + k$ a’s and $p + m$ b’s, but only $p$ c’s.
  • So $uv^2xy^2z \notin L$.
  • This is a contradiction.

• Cases 3, 4, and 5 are similar.

• Therefore, $L$ is not context-free.
The Idea Behind the Proof

- If a CFL contains a string \( w \) with a sufficiently long derivation

\[
S \Rightarrow w,
\]

then some variable \( A \) must appear more than once in the derivation.

- That is, we must have

\[
S \Rightarrow uAz \Rightarrow uvAy \Rightarrow uvxyz,
\]

for some strings \( u, v, x, y, \) and \( z \).
The Idea Behind the Proof

• Thus, $A \Rightarrow^* vAy$ and $A \Rightarrow^* x$.
• We may repeat the derivation

$$A \Rightarrow^* vAy$$

as many times as we like (including zero times), producing strings $uv^nxy^n z$, for any $n \geq 0$. 
The Proof

Proof.

• Let $b$ be the largest number of symbols on the right-hand side of any grammar rule. (Assume $b \geq 2$.)
• Let $h$ be the height of the derivation tree of a string $s$.
• Then $s$ can contain at most $b^h$ symbols.
• Equivalently, if $s$ contains more than $b^h$ symbols, then the height of the derivation tree of $s$ must be more $h$. □
Proof.

- Now $|V|$ is the number of variables in the grammar of $L$.
- So if a string in $L$ has a length greater than $b^{|V|+1}$, then the height of its derivation tree must be more than $|V| + 1$.
- So let $p = b^{|V|+1}$ and suppose that a string $s \in L$ has length at least $p$. 
Proof.

- Consider the longest path through the derivation tree of $s$.
- It has length at least $|V| + 1$.
- That path has $|V| + 2$ nodes on it, counting the root node $S$ and the leaf node, which is a terminal.
The Proof

Proof.

- Thus, $|V| + 1$ of the nodes are variables.
- So one of them must be repeated.
- As we follow the longest path back from leaf to root, let $A$ be the first variable that repeats.
- Now consider these two occurrences of $A$ along the longest path.

□
The Proof
The Proof

Proof.

- The “middle part” of this tree, the part that produces

\[ A \Rightarrow \forall A y, \]

may be repeated as many times as desired.
The Proof
The Proof

Proof.

• Therefore, the strings $uv^2xy^2z$, $uv^3xy^3z$, etc. can also be derived.
• So can the string $uxz$.
• Furthermore, we may assume that this was the shortest derivation of $s$.
• It follows that $v$ and $y$ cannot both be empty strings.
• If they were, then the middle part of the derivation would be

$$A^* \Rightarrow A,$$

which could be eliminated.
• Thus, $|vy| > 0$. 

□
Finally, we must show that $|vxy| \leq p$.

The subtree rooted at the second-to-last $A$ has height at most $|V| + 1$.

So the string $vxy$ has at most $b^{|V|+1} = p$ symbols.

□
**Example**

- Let $\Sigma = \{a, b\}$.
- Show that the language

$$\{ww \mid w \in \Sigma^*\}$$

is not context-free.
- Use $s = a^p b^p a^p b^p$. 

An Example
Consider the grammar for the language \( L = \{ a^n b^n \} \)

1. (5pts) Chomsky-ize this grammar
2. (5pts) Find all derivation trees that do not have self-embedded non-terminals

2. (5pts) Why does the pumping lemma argument not show the language PALINDROME is not context free? Show how \( v \) and \( y \) can be found such that \( w = uv^ny^nz \) are also in PALINDROME no matter what \( w \) is.

3. (5pts) How would you go about proving the following theorem? If \( L \) is a language over the one-letter alphabet \( \Sigma = \{ a \} \) and \( L \) can be shown to be non-regular using the pumping lemma for regular languages, then \( L \) can be shown to be non-context-free using the pumping lemma for context-free languages.